

Mapeando Cenário & LEL para o modelo Statechart: Uma Estratégia para Verificação da Especificação de Requisitos

Silvia Pereira Azevedo Sousa, Carla Taciana L. L. Silva

Universidade Federal de Pernambuco (UFPE) – Recife/PE – Brasil

spas@cin.ufpe.br, ctlls@cin.ufpe.br

Abstract. *The Software Requirements Specification (SRS) phase is critical in the software development process. The use of scenarios is quite popular for specifying requirements and is usually described in natural language (NL), giving chance for ambiguous, incomplete and inconsistent requirements. These problems are concerning, as the NL specifications may impair the final quality of the software. In this sense, a technique based on Scenario & LEL (Language Extended Lexicon) - C & L - may be an appropriate solution to reduce these problems, since it is scenario technique controlled by LEL, which in turn is a glossary that defines a set of words that belong to a domain. Although C & L is a controlled NL, it does not guarantee the completeness and consistency of the requirements. There is a need to verify these specifications through a mapping to a more formal language. This work proposes a C & L mapping for the statechart model with the objective of verifying the completeness and consistency of the requirements specified in the scenarios. A scenario of sending orders to providers was used as an example to illustrate the application of the defined mapping rules. As a result, it was observed the approach potential to aid the verification of requirement described in NL.*

Resumo. *A fase de especificação de requisitos de software (SRS) é fundamental no processo de desenvolvimento de software. O uso de cenários é bastante popular para especificar requisitos e, normalmente, são descritos em linguagem natural (LN), dando margem a requisitos ambíguos, incompletos e inconsistentes. Esses problemas são preocupantes, pois as especificações em LN podem prejudicar a qualidade final do software. Nesse sentido, uma técnica baseada em Cenário & LEL (Language Extended Lexicon) - C&L - pode ser a solução adequada para reduzir esses problemas, pois é uma técnica de cenários controlada pelo LEL, que por sua vez é um glossário que define um conjunto de palavras que pertencem a um domínio. Apesar de C&L ser uma LN controlada, ela não garante a completude e consistência dos requisitos. Surge a necessidade de verificar essas especificações por meio de um mapeamento para uma linguagem mais formal. O presente trabalho propõe um mapeamento de C&L para o modelo de statechart com o objetivo de verificar a completude e consistência dos requisitos especificados nos cenários. Foi utilizado um cenário de envio de pedidos a fornecedores como exemplo ilustrativo de aplicação das regras de mapeamento definidas. Como resultado, observou-se o potencial da abordagem para a verificação de requisitos descritos em LN.*

1. Introdução

Na Engenharia de Requisitos (ER) existem técnicas que auxiliam o engenheiro de requisitos a compreender o universo do problema como, por exemplo, as técnicas baseadas em cenários (casos de uso, storyboard, C&L, etc) que têm sido muito utilizadas para entender, modelar e validar os requisitos de usuários.

Segundo Santander (2002), cenários são utilizados tanto para a elicitación de requisitos, como também para a sua validação e especificação. Os cenários podem ser escritos como texto em linguagem natural (LN), complementado por diagramas, capturas de tela, etc. (Kotonya e Sommerville, 1998).

Para facilitar o processo de comunicação entre as partes interessadas, é necessário fazer uso de uma linguagem natural. Segundo Sarmiento, Leite e Almentero (2014), a LN é amplamente utilizada na descrição de software, já que proporciona um canal de comunicação entre os intervenientes no processo de construção de software (usuários, clientes desenvolvedores, testadores). Além disso, a LN é facilmente associada com métodos e linguagens para especificação de requisitos, tais como os cenários apresentados por Leite et al. (2000) e o uso de léxicos apresentado por Leite e Franco (1993).

A abordagem C&L (Cenário & LEL), segundo Leite et al., (1997), contribui no desenvolvimento de cenários mais controlados, efetivos e não ambíguos, diferenciando-se da abordagem de casos de uso por considerar uma visão mais abrangente, incluindo a descrição de cenários e informações sobre o contexto no qual o sistema será inserido.

No entanto, especificações de requisitos baseadas em LN, como o C&L, necessitam ser verificadas ainda na fase inicial da produção do software, visando melhorar a qualidade do produto. Essa verificação nas fases iniciais contribui para redução de falhas e redução do custo de manutenção depois que o produto final for entregue (Sarmiento, Leite e Almentero, 2014).

Os autores Ryser e Glinz (1999) afirmam que mapear cenários para técnica de statecharts ajuda a encontrar muitas omissões, ambiguidades e inconsistências nas especificações. Para Harel (1987), a técnica de statecharts permite gerar diagramas compactos e bastante expressivos, ou seja, podem representar comportamentos complexos de maneira estruturada permitindo visualizar descrições em diferentes níveis de detalhamento, corroborando para especificações mais gerenciáveis e compreensíveis.

Motivado por este contexto, este trabalho propõe um mapeamento de C&L para statechart, com o intuito de promover uma verificação nos cenários, visando identificar problemas de incompletude e inconsistência mais cedo, corroborando para uma especificação de requisitos com melhor qualidade. Outro benefício é permitir aos analistas de requisitos uma compreensão mais aprofundada no domínio do sistema a ser desenvolvido já na fase inicial, pois precisarão compreender com mais detalhes o cenário para poder construir o modelo de statechart. Assim, espera-se identificar uma quantidade maior de problemas nos requisitos já nessa fase.

O trabalho está organizado da seguinte maneira. Na Seção 2, será apresentado o background. Na Seção 3, será demonstrado o mapeamento de C&L para statecharts. Na seção 4, será apresentado um exemplo ilustrativo. A Seção 5 descreve os trabalhos

relacionados. E por último, a Seção 6 expõe as conclusões e os direcionamentos para trabalhos futuros.

2. Background

Esta seção descreve a abordagem C&L (Cenário & LEL), em seguida apresenta o modelo statecharts e por fim define um mapeamento de C&L em conformidade com o modelo statecharts com o objetivo de verificar a qualidade da especificação dos requisitos descritos nos cenários.

2.1 Abordagem C&L

Esta abordagem se concentra em cenários e utiliza um vocabulário controlado com base no léxico da aplicação, o LEL. Segundo Zorman (1995), cenário é uma descrição parcial do comportamento da aplicação que ocorre em um dado momento em um contexto geográfico específico - uma situação. Para Leite e Franco (1993), o LEL é um glossário que define um conjunto de palavras que pertencem a um domínio. Ele está ligado a uma ideia simples: “understand the language of the problem, without worrying about understanding the problem”.

Segundo Leite et al., (1997), C&L pode ser utilizado na fase da elicitação de requisitos como também ao longo de todo o processo de desenvolvimento do software permitindo a validação com o cliente/usuário. O principal objetivo do léxico é capturar o vocabulário da aplicação e sua semântica, adiando a compreensão da funcionalidade da aplicação (Vilela, 2015).

A linguagem natural baseada em cenário utilizada neste trabalho é a mesma definida por Sarmiento, Leite e Almentero (2015). Este cenário tem como elementos principais título, objetivo, contexto, recursos, atores, episódios, pré e pós-condição, exceções e restrições, detalhados abaixo:

1. O título - identifica o cenário utilizando uma sentença declarativa e deve ser único;
2. Objetivo - descreve a proposta do cenário usando uma sentença declarativa, dando uma ideia geral sobre o cenário principal;
3. Contexto – descreve o estado inicial do cenário usando sentenças declarativas, deve ser descrito através de pelo menos um dos seguintes sub-componentes: pré-condição, pós-condição, localização geográfica ou temporal;
4. Recurso – é uma enumeração de entidades passivas utilizadas pelos atores nos episódios para alcançar o objetivo do cenário. Os recursos devem aparecer em pelo menos um dos episódios;
5. Atores – são uma enumeração de entidades ativas diretamente envolvidos com a situação. Os atores devem aparecer em pelo menos um dos episódios;
6. Episódios - são um conjunto de ações que dão uma descrição operacional do comportamento. Eles representam o fluxo principal, que é uma sequência de etapas em que tudo funciona conforme o esperado;
7. Pré-condição e a pós-condição – são descritos como sentenças declarativas envolvendo estados (ator/ recursos/sistemas relevantes), eles são diferentes da

pré-condição do contexto e pós-condição do contexto, pois respectivamente estão relacionados com o estado do sistema antes que o cenário é iniciado e quando os episódios são realizados;

8. Exceções - são situações que impedem o andamento normal do cenário. Seu tratamento deve ser feito usando uma sentença. Ele representa as alternativas / fluxos excepcionais;
9. Restrições - descrevem aspectos não funcionais restringindo a qualidade dos atributos dos recursos, episódios ou sub- componentes do contexto.

O LEL é construído por termos chamados de símbolos e são definidos por dois atributos: noção e resposta comportamental. A noção descreve o símbolo no sentido denotativo, ou seja, o significado dele dentro do contexto. Já a resposta comportamental descreve o símbolo no sentido conotativo, ou seja, como o símbolo se comporta e quais os efeitos que causa no sistema. O LEL também apresenta uma categorização para ser aplicadas nos símbolos, com o intuito de orientar o engenheiro de requisitos durante a descrição dos requisitos. As categorias são: Sujeito, Objeto, Verbo e Estado.

O sujeito – são elementos ativos que executam ações; Objeto – são elementos passivos em que os sujeitos executam ações; Verbo - são ações que os sujeitos executam nos objetos; Estado – são situações em que os sujeitos e os objetos podem estar.

Contudo, apesar de C&L ser uma técnica de cenários controlada pelo LEL, ela não é suficiente para garantir completude e consistência dos requisitos. Daí a necessidade de verificar essas especificações por meio de um mapeamento para uma linguagem mais formal de forma a ajudar os desenvolvedores a exercitar os diferentes fluxos de execução do software, ajudando a melhorar a qualidade do produto ainda na fase de especificação de requisitos do software.

2.2 Statecharts

Segundo Harel (1987), statecharts são formalismos visuais que podem ser usados para especificar comportamentos arquiteturais do sistema garantindo que as especificações sejam mais completas. Essa técnica permite gerar diagramas compactos e bastante expressivos, ou seja, podem representar comportamentos complexos de maneira estruturada permitindo visualizar descrições em diferentes níveis de detalhamento corroborando para especificações mais gerenciáveis e compreensíveis. Os principais elementos do statecharts são: Estados, Transições, Eventos, Ações e Regiões. Os estados são interações ocorridas nos objetos para que uma condição seja satisfeita e a ação executada ou quando aguarda um evento, além disso, os estados são representados como caixas e as transições são representadas entre estados como flechas.

As transições capturam uma mudança de estado causado pela ocorrência de um acontecimento associado. A transição pode ser guardada por alguma condição, representada por um nome de condição ou uma expressão fechada entre colchetes. Um evento capta uma condição necessária para o disparo da transição.

Uma ação é uma operação auxiliar associada a uma transição de estado que é aplicada quando a transição está ativada. O paralelismo é representado pela divisão de um estado composto em regiões que são mostrados e separados por linhas tracejadas em Pimentel et al. (2014). Os estados podem ser estruturados em uma hierarquia (super-Estado e sub-Estado). Estados ficam dentro de regiões e um super- Estado deve ter uma região dentro

para poder ter os sub - estados. As regiões são usadas para organizar os modelos e principalmente para tornar estados paralelos. Super-Estado são chamados "Estados composites" - ele pode ser adicionado da mesma forma que os estados regulares.

3. Mapeando C&L para Statecharts

Com o intuito de mostrar a conformidade existente entre os elementos de C&L e Statecharts, foi elaborada uma tabela que define regras de mapeamento (Tabela 1) para derivar Statecharts a partir de C&L, no qual o objetivo é verificar a qualidade das especificações através de uma linguagem formal.

A Tabela 1 está organizada contemplando os elementos de C&L e do statecharts respectivamente, além da representação gráfica como uma contribuição. Na sequência, vamos apresentar como os conceitos de C&L podem ser acomodados dentro do Statecharts, a fim de representar o comportamento do sistema.

Como explicado na seção 3.1, em C&L o título identifica o cenário utilizado e deve ser exclusivo. Na seção 3.2, explicamos que em statecharts, o super-estado representa o estado maior, delimitando a aplicação do sistema. Portanto, na linha (1) da Tabela 1 o título é mapeado como super-estado em statecharts (Tabela 1).

O episódio de sentença simples em C&L representa o fluxo principal necessário para completar o cenário, representado por uma sequência de sentenças onde tudo funciona como esperado. Cada sentença é um episódio pode ser visto como uma tarefa. No statechart, um estado é representado pela execução de tarefa. A realização de uma tarefa (estado) conduz a uma transição para outro estado que representa outra tarefa a ser executada na sequência. Assim, uma sentença do episódio simples é mapeada como um estado em statechart (linha (2) da Tabela 1).

Tabela 1. Regras de mapeamento

	Elementos C&L	Elementos Statecharts
1	Título	Super-Estado
2	Episódio sentença Simples	Estado (Tarefas sequenciais)
3	Resultado da execução da sentença	Evento da Transição
4	Episódio Sentença condicional	Estado XOR Transição de um episódio de sentença simples para outro episódio de sentença excepcional.
5	Episódio Sentença Loop	Estado
		Transição com repetições no mesmo estado
6	Episódio Sentença Opcional	Evento de transição (tarefa opcional pode ser executada repetidamente ou possivelmente pode não ser executada).
7	Episódio Sentença	Estado ortogonal (Estado AND) – vários
8	Pré e Pós condição, Restrição	Annotations

Fonte: o autor

O resultado da sentença em C&L representa o resultado após a execução da tarefa representada na sentença. Já no statechart, a conclusão de um estado causa uma transição para outro estado. Note que a transição utiliza eventos e/ou condições para que um estado mude para o outro. O evento é o resultado da conclusão do estado e a condição é algo que deve ser verdadeiro, ou então a transição não será executada. Sendo assim, o resultado de uma sentença é mapeado para o evento de uma transição no statechart (linha (3) da Tabela 1).

O episódio de sentença condicional em C&L é representado por um fluxo de exceção que apresenta sentenças alternativas que dependem de uma condição interna ou externa para serem executadas. No statechart, o estado XOR representa um fluxo alternativo permitindo estados alternativos. Assim, a transição parte de um estado para estados alternativos diferentes a depender da condição satisfeita. Desse modo, uma sentença condicional é mapeada como um estado destino da transição XOR no statechart (linha (4) da Tabela 1).

O episódio de sentença loop em C&L identifica repetições na estrutura da sentença até que a condição seja satisfeita. Em statecharts, um loop é representado como uma transição que se repete em um mesmo estado até que uma condição seja satisfeita. Desta maneira, a sentença loop é mapeada como um estado que possui uma transição apresentando uma condição para que o evento seja modificado e a tarefa (estado) seja concluída no statechart (linha (5) da Tabela 1).

O episódio de sentença opcional em C&L identifica ações que podem ou não serem executadas. Em statechart, uma execução opcional dentro de uma sequência, possivelmente uma das tarefas (estados) podem não ser executadas sempre, podem ser executadas zero ou mais vezes ou podem ser executados uma ou mais vezes. Assim, o episódio de sentença opcional é mapeado como uma tarefa opcional no statechart (linha (6) da Tabela 1).

Sentenças concorrentes em C&L ocorrem paralelamente e são limitadas pelo símbolo “#”. No statechart, o estado ortogonal (Estado AND), representa vários estados executando paralelamente. Desta forma, sentenças concorrentes são mapeadas como estados ortogonais / paralelos em statechart (linha (7) da Tabela 1).

Em C&L as pré, pós-condições e as restrições são descritas como sentenças declarativas envolvendo aspectos não funcionais que corroboram para a qualidade do sistema. Esses atributos são importantes na modelagem de sistemas concorrentes (Leite e Franco, 1990). No statechart, as informações importantes do sistema como pré-condições, dados (entrada, saída e intervalos) e RNF (Requisito Não-Funcional) são capturados em Annotations. Desta maneira, na (linha (8) da Tabela 1) as pré, pós-condições e as restrições são mapeadas como Annotations no statechart.

4. Exemplo Ilustrativo

Os modelos de origem apresentados nas Figuras (1 e 2) foram retirados do trabalho de Sarmiento, Leite e Almentero (2015). No cenário “Enviar pedido” são identificados 15 eventos (13 fluxos principais – episódios e 2 fluxos excepcionais). Neste cenário são descritos interações entre o Sistema Broker Online e os serviços de parceria com fornecedores (FornecedorA, FornecedorB e FornecedorC) representados pelos episódios 10, 11 e 12 usando a linguagem do cenário. O cenário “Enviar pedido” interage com os

episódios 10, 11 e 12 que são executados paralelamente em uma ordem não sequencial limitada pelo símbolo “#” como, por exemplo, o episódio 10 que foi detalhado na Figura 2 (os episódios 11 e 12 faz uso do mesmo cenário).

O modelo statechart completo é gerado a partir do cenário “Enviar pedido” (modelo de origem) apresentado na Figura 1. As sentenças simples dos episódios representados através de identificadores anotados no modelo de statechart em conformidade com o cenário que corresponde aos estados no statechart: E1 (O cliente carrega a página de login), E2 (O sistema Broker solicita as informações de login do cliente), E3 (O cliente insere suas informações de login), E4 (O sistema Broker verifica as informações de login fornecidas), E5 (O sistema Broker exibe uma página de pedido), E6 (O cliente cria um novo pedido), E8 (O cliente envia p pedido), E9 (O sistema Broker transmitiu o pedido aos fornecedores), E13 (PROCESSO DE OFERTAS).

Figura 1. Modelo C&L (Descrição do cenário "Enviar Pedido" no Sistema Broker)

TÍTULO: Enviar pedido
OBJETIVO: permitir que os clientes encontrem o melhor fornecedor para um determinado pedido.
CONTEXTO:
PRE-CONDIÇÃO: O Sistema Broker está online e a página de boas-vindas do sistema Broker está sendo exibida.
ATOR: Cliente, Sistema Broker
RECURSOS: página de login, informações de login, pedido
EPISÓDIOS
1. O Cliente carrega a página de login
2. O sistema Broker solicita as informações de login do Cliente
3. O Cliente insere suas informações de login
4. O sistema Broker verifica as informações de login fornecidas
5. O sistema Broker exibe uma página de pedido
6. O Cliente cria um novo pedido
7. **REPETIR** o cliente adicionar um item ao pedido **ENQUANTO** o cliente possui mais itens para adicionar ao pedido.
8. O Cliente envia o pedido
9. O sistema Broker transmitiu o pedido aos Fornecedores. **Pós-condição:** um pedido tem que ser transmitido
10. # O FORNECEDORA FAZ OFERTA PARA O PEDIDO
11. O FORNECEDOR B FAZ OFERTA PARA O PEDIDO
12. O FORNECEDOR C FAZ OFERTA PARA O PEDIDO #
13. PROCESSO DE OFERTAS.
EXCEÇÕES
4.1 Se as informações de login do Cliente não forem precisas, **ENTÃO** o Sistema Broker exibirá uma mensagem de alerta
8.1 Se o pedido estiver vazio, então, o sistema Broker exibirá uma mensagem de erro.

Fonte: Sarmiento, Leite e Almentero (2015)

Figura 2. Cenário FornecedorA oferta pedido

TÍTULO: Oferta do Fornecedor para pedido
OBJETIVO: O FornecedorA fez uma oferta
CONTEXTO: criar uma oferta para um pedido
PRE-CONDIÇÃO: um pedido foi transmitido
POS-CONDIÇÃO: o FornecedorA fez uma oferta
ATOR: FornecedorA, Sistema Broker
RECURSOS: Pedido, Oferta
EPISÓDIOS
1. O fornecedorA recebe um pedido e examina
2. O FornecedorA envia uma oferta para o pedido
3. O Sistema Broker envia a Oferta ao Cliente
EXCEÇÕES
1.1 SE o FornecedorA não pode satisfazer o pedido ENTÃO o fornecedor A avança o pedido

Fonte: Sarmiento, Leite e Almentero (2015)

As sentenças condicionais dos episódios são representados através de identificadores anotados no modelo de statechart: E4.1 (Se as informações de login não forem precisas, ENTÃO o sistema Broker exibirá uma mensagem de alerta), E8.1 (Se o pedido estiver vazio, então o sistema Broker exibirá uma mensagem de erro); A sentença de loop é representada pelo identificador: E7 (REPETIR o cliente adiciona um item ao pedido ENQUANTO o cliente possuir mais itens para adicionar ao pedido) e as sentenças concorrentes são representadas pelos identificadores: E10 (FornecedorA faz oferta para o pedido), E11 (FornecedorB faz oferta para o pedido), E12 (FornecedorA faz oferta para o pedido).

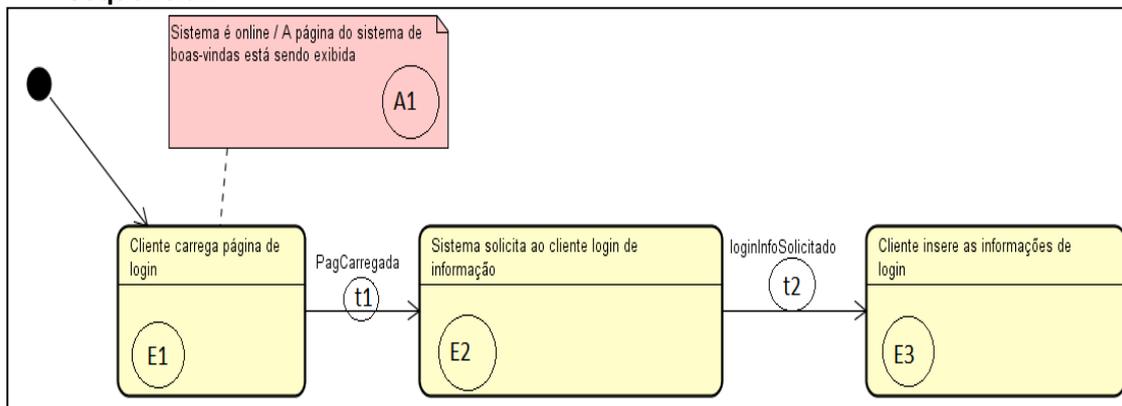
No C&L, o resultado da execução de uma sentença é mapeado para statechart como um evento de transição. As transições podem apresentar eventos e/ou condições, vai depender do tipo de sentença. Os eventos das transições são representados através dos identificadores t1 (paginaCarregada), t2 (loginInfoSolicitado), t3 (loginInfoInserido), t4 (loginVerificado), t5 (alertaMsgExibida), t6 (pedidoExibidoPagina), t7 (novo PedidoCriado), t8 (itemsAdicionadoPedido), t9 (pedidoSubmetido), t10 (erroMsgExibida), t11 (pedidoTransmitido), t12 (pedidoOfertado), t13 (processoOfertado).

As condições são representadas pelos indicadores c1 (loginInfoPreciso), c2 (loginInfoNaoEpreciso), c3 (clienteTemMaisItem), c4 (pedidoNaoVazio), c5 (pedidoEvazio), c6 (pedidoSatisfeito), c7 (pedidoNaoSatisfeito). Adicionalmente, são criadas anotações (Annotations) que são representadas pelos identificadores A1 (Sistema é online / A página do sistema de boas-vindas está sendo exibida), A2 (Pedido tem sido/ transmitido), A3 (Pedido tem sido transmitido / FornecedorA fez uma oferta), A4 (Pedido tem sido transmitido / FornecedorB fez uma oferta), A5 (Pedido tem sido transmitido / FornecedorC fez uma oferta) nos modelos statecharts que são informações importantes que servirão para refinar a qualidade do sistema.

A construção do statechart se deu através das regras de mapeamento da (Tabela 1), que foram criadas com o intuito de mapear o comportamento do sistema representado no modelo C&L (Figura 1) para um modelo mais estruturado e formal.

Na Figura 3, o sistema é iniciado com três estados (E1, E2 e E3) sendo executados no fluxo principal de maneira sequencial. No cenário, estes elementos estão representados como sentenças simples, ou seja, sequência de sentenças onde tudo funciona como esperado. Este mapeamento estar conforme a regra definida na linha 2 da Tabela 1. Sabe-se que, a cada mudança de estado ocorre uma transição (t1 e t2), que se dá quando um evento ocorre e esses eventos são os resultados da conclusão do estado. Essa execução do sistema está de acordo com a regra definida na linha 3 da Tabela 1.

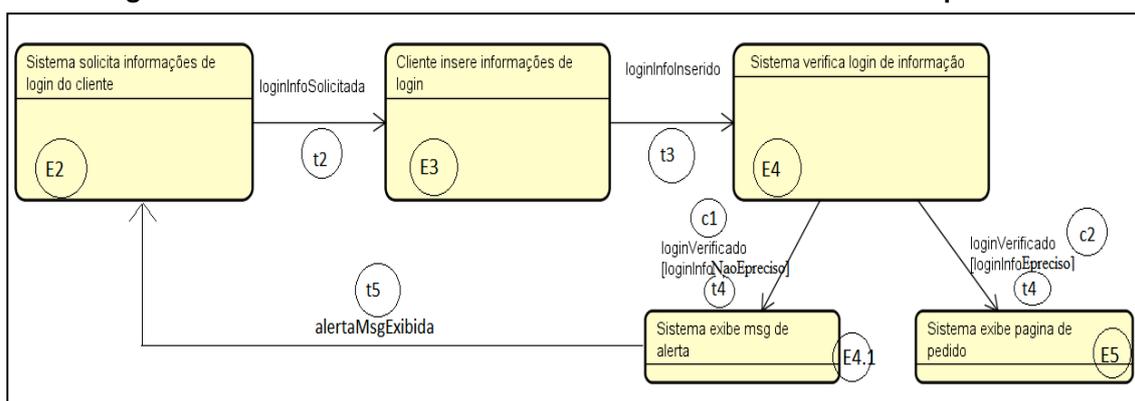
Figura 3. Estados simples sendo executados no fluxo principal de maneira sequencial.



Fonte: O autor

Na Figura 4, os estados E2 e E3 estão sendo executados no fluxo principal de maneira sequencial. No E4 ocorre uma transição de um episódio de sentença simples para outro episódio de sentença excepcional (E4.1 ou E5). Se ocorrer falha em E4.1 e a condição c1 não for atendida, o sistema dispara o evento t5 e retorna para o estado E2, caso contrário o sistema dispara o evento t4 e muda para o estado E5. Essa regra está definida na linha 4 da Tabela.

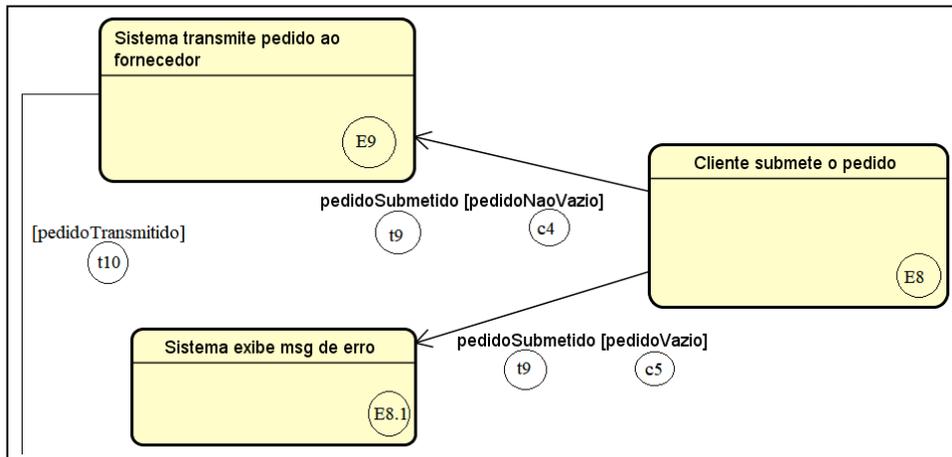
Figura 4. Estado alternativo sendo executado de maneira não- sequencial.



Fonte: O autor

Na Figura 5, no estado E8 ocorre uma transição de um episódio de sentença simples para outro episódio de sentença excepcional (E8.1 ou E9). Se ocorrer falha em E8 e a condição c5 for atendida, o sistema dispara o evento t9 e lança uma mensagem de erro E8.1, caso contrário, se a condição c4 for atendida, o sistema dispara o evento t9 e muda para o estado E9. Essa regra está definida na linha 4 da Tabela 1.

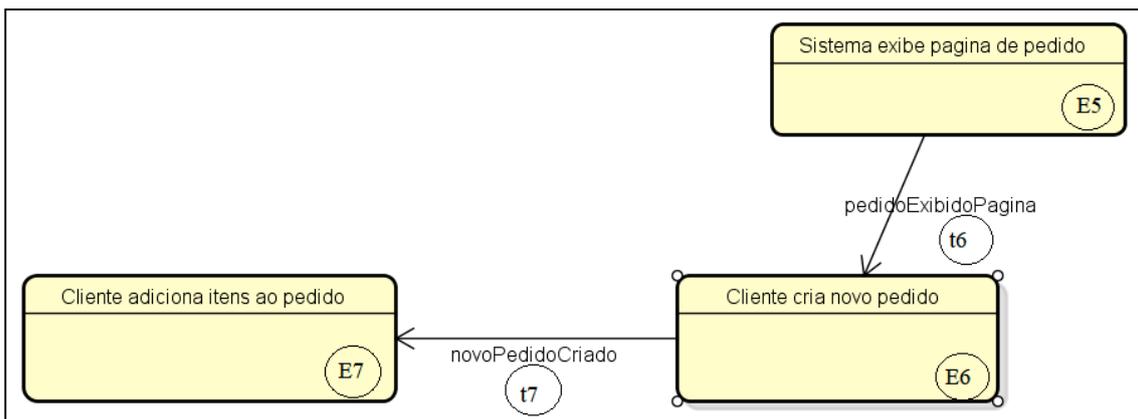
Figura 5. Estado alternativo sendo executado de maneira não- sequencial.



Fonte: O autor

Na Figura 6, os estados E5, E6 e E7 são executados de forma sequencial simples no fluxo principal. A cada mudança de estado ocorre um evento gerando uma transição. Assim, quando o sistema dispara a transição t6, ocorre a mudança do estado E5 para o estado E6. O estado E6 é executado quando o evento t7 ocorre, ocasionando a mudança para o estado E7. Este passo recai sobre a regra definida na linha 3 da Tabela 1.

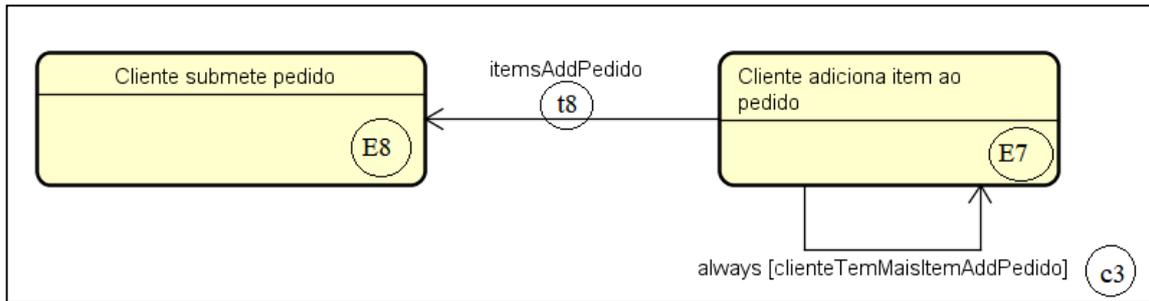
Figura 6. Estados simples sendo executados no fluxo principal de maneira sequencial.



Fonte: O autor

Na Figura 7, o estado E7 apresenta uma transição com repetições no mesmo estado. Esta repetição é representada através de um loop, ou seja, sempre que a condição c3 for atendida, a tarefa continuará sendo executada até que a condição não seja mais satisfeita e o evento t8 ocorra, havendo a mudança para o estado E8. A regra definida nesta execução está em conformidade com a linha 5 da Tabela 1.

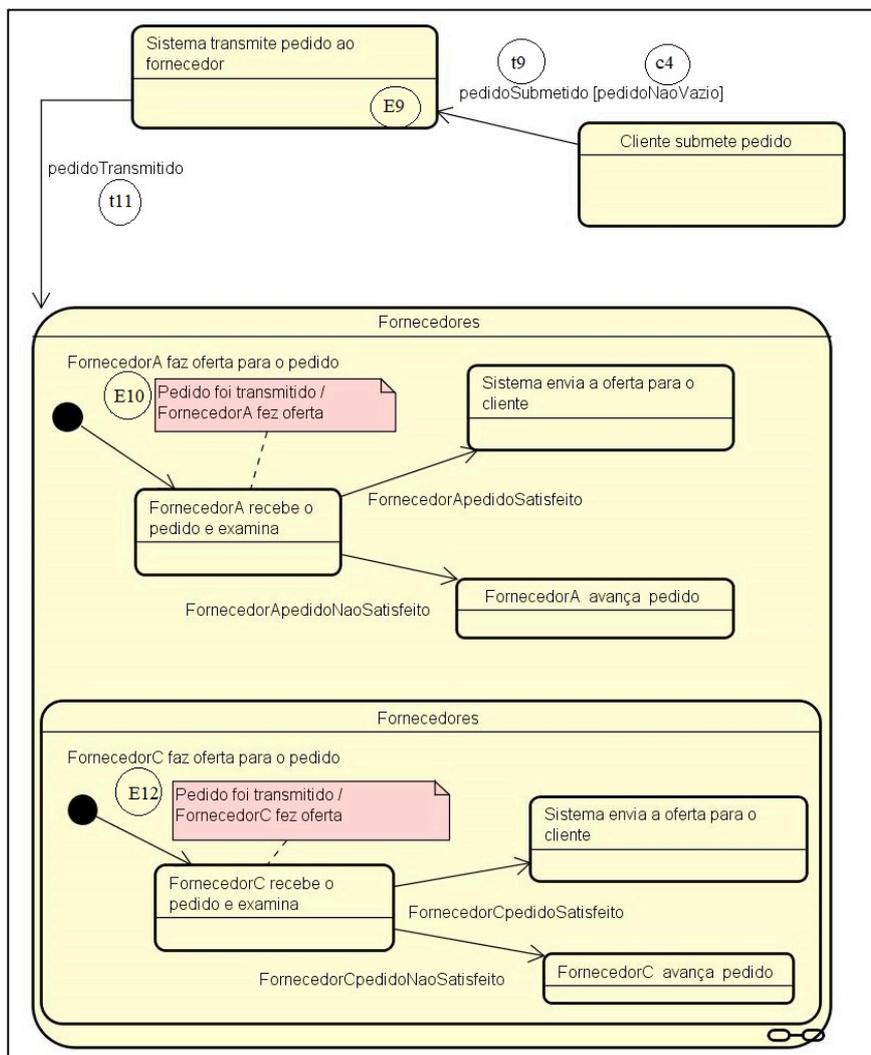
Figura 7. Estado loop sendo executado com repetições no mesmo estado.



Fonte: O autor

Na Figura 8, no estado E9 ocorre uma interação com um estado ortogonal. Esses estados (E10, E11 e E12) são executados paralelamente no sistema. Assim, quando o evento t11 é disparado, a execução do estado E9 é concluída, ocorrendo a mudança para o estado ortogonal. A regra aplicada nesta execução está em concordância com a linha 7 da Tabela 1.

Figura 8. Interação do estado E9 com o estado ortogonal.



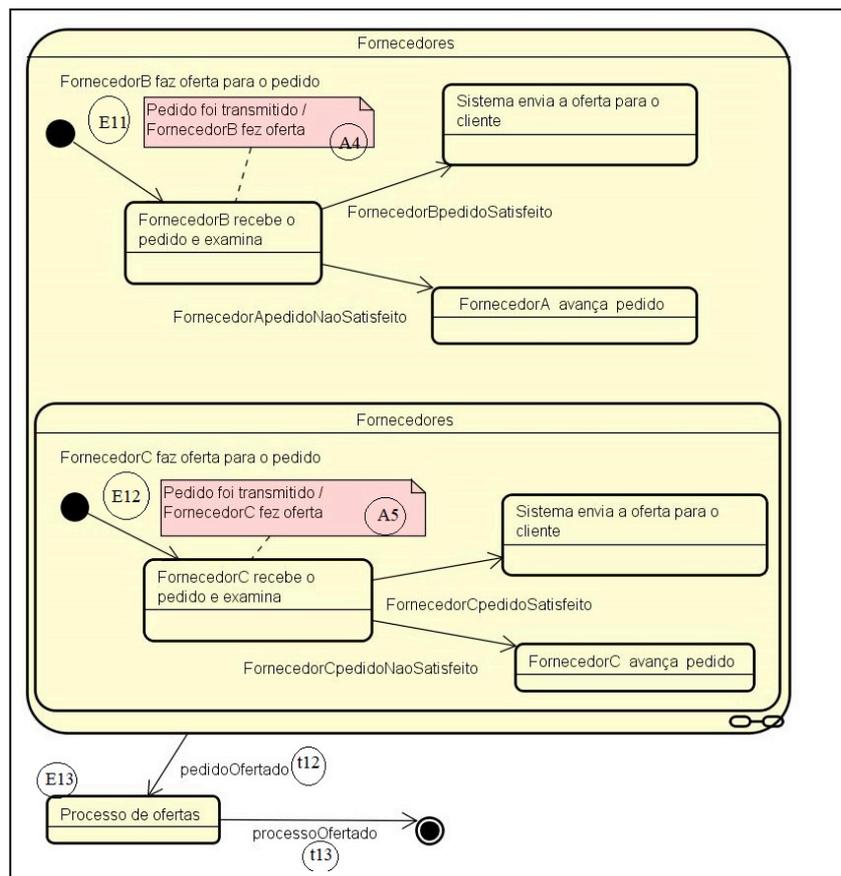
Fonte: O autor

Dentro de cada estado ortogonal acontecem execuções em paralelo dentro do sistema como, por exemplo, os estados E10, E11 e E12, nos quais ocorrem transições nos episódios de sentença simples para outro episódio de sentença excepcional que devem ser representadas pelo estado XOR. A descrição da execução desses estados é a mesma utilizada nos estados E4 e E8.

Para finalizar a saída do sistema, o evento de transição t12 é disparado para que possa ocorrer a modificação dos estados ortogonais para o estado E13, que é o último no fluxo principal do cenário. Quando o evento t13 é disparado, o estado E13 é concluído e o sistema findado (Figura 9).

Como um adicional, para exemplificar, no modelo statechart foi inserido os identificadores A4 e A5, respectivamente nos estados E11 e E12 em forma de anotações (Annotations), com o intuito de trazer informações importantes que poderão contribuir no refinamento da qualidade do sistema. A regra utilizada está conforme a linha 8 da Tabela 1.

Figura 9. Modificação do estado ortogonal para o estado E13.



Fonte: O autor

5. Trabalhos Relacionados

Algumas pesquisas têm mostrado a importância de verificar cenários descritos em LN, através de uma linguagem mais formal, como os trabalhos de Sarmiento, Leite e Almentero (2015) e Ryser e Glinz (1999). Em Sarmiento, Leite e Almentero (2014), os

requisitos iniciais são definidos como cenários e estes cenários são transformados em diagramas de atividades de maneira automatizada através do uso da ferramenta C&L para apoiar testes automatizados. As trajetórias do diagrama de atividades são geradas a partir das interações entre os episódios, exceções e restrições e são utilizados para a geração de cenários de teste iniciais através do uso de um gráfico de pesquisa e estratégia de combinação de trajetória. Uma limitação deste trabalho é em relação a geração de teste de exceções e requisitos não funcionais e também esta abordagem não é aplicável a situações que envolvem simultaneidade. O diferencial dessa abordagem é que podemos usar as sentenças não sequencias do cenário para representar situações de simultaneidade usando o Estado AND. Podemos também capturar os RNFs através de Annotations e representá-los no modelo statecharts.

Em Sarmiento, Leite e Almentero (2014), os requisitos iniciais são definidos como cenários e estes cenários são transformados em diagramas de atividades de maneira automatizada através do uso da ferramenta C&L para apoiar testes automatizados. As trajetórias do diagrama de atividades são geradas a partir das interações entre os episódios, exceções e restrições e são utilizados para a geração de cenários de teste iniciais através do uso de um gráfico de pesquisa e estratégia de combinação de trajetória. Uma limitação deste trabalho é em relação a geração de teste de exceções e requisitos não funcionais e também esta abordagem não é aplicável a situações que envolvem simultaneidade. O diferencial da nossa abordagem é que podemos usar as sentenças não sequencias do cenário para representar situações de simultaneidade usando o Estado AND, podemos também capturar os RNFs através de Annotations e representá-los no modelo statecharts.

Outro trabalho que possui similitude com o nosso é o descrito por Sarmiento, Leite e Almentero (2015), que apresenta uma abordagem baseada em Petri-net com o objetivo de analisar de maneira eficaz os cenários que descrevem os requisitos de entrada de um sistema. Para permitir uma análise automatizada, os cenários e suas interações são traduzidas em conformidade com o metamodelo Petri-net através de um procedimento sistemático (mapeamento de cenário para elementos Petri-Net), além de utilizar um algoritmo para transformar de cenário para Petri-Net e um algoritmo de integração de Petri-nets. A nossa abordagem diferencia-se desta, pois define regras de mapeamento para transformar a abordagem C&L em modelo statechart.

Uma abordagem bastante correlata com a nossa pesquisa é o trabalho de Ryser e Glinz (1999) que faz uso de uma abordagem baseada em cenários que valida e testa o sistema através do uso de statechart. Essa validação de cenário é acompanhada por medidas de verificação. A verificação é apoiada por um método (SCENT-Method) de formalização: Ao converter cenários de linguagem natural em statecharts, muitas omissões, ambiguidades e inconsistências podem ser encontradas. Uma limitação que este trabalho apresenta é em relação ao processo de formalização, ou seja, o mapeamento de ações em cenários de linguagem natural para estados ou transições não é bem definida e clara. Como consequência, um cenário transformado em statechart por um desenvolvedor difere significativamente de um statechart desenvolvido a partir do mesmo cenário por outro desenvolvedor. Diferentemente deste trabalho, a nossa abordagem faz uso de regras de mapeamento que permitem transformar o C&L em statecharts, permitindo que os modelos de statechart obtidos por parte de desenvolvedores distintos sejam muito semelhantes ou iguais.

Entretanto, existem outras abordagens que também derivam statecharts de modelos de requisitos como, por exemplo, a abordagem do autor Pimentel et al., (2014) que captura os requisitos de entrada de um sistema através do Goal Model (modelo de objetivo). O intuito é representar o comportamento do sistema e refinar os requisitos (elementos de design) originais por considerar alternativas comportamentais levando a projetar novas soluções e em seguida derivar este mesmo modelo para statecharts.

Uma limitação é que esta abordagem não apoia o desenvolvimento de sistemas auto-adaptativos. Além disso, a abordagem não considera cenários na especificação de requisitos, diferentemente da nossa.

O diferencial desta abordagem em relação as supracitadas, é que a medida que os requisitos estão sendo capturados do cenário para serem modelados em statecharts, o artefato já está sendo verificado, pois espera-se que o analista de requisitos consiga detectar um maior número de problemas relacionados a inconsistência e incompletude do que quem analisou o cenário sem a preocupação de elaborar o modelo statechart. Resultados de uma avaliação inicial da abordagem demonstraram esses benefícios. Essa avaliação contou com a participação de alunos do curso de graduação de Ciência da Computação, Engenharia da Computação e Sistema de Informação. A amostra foi dividida em dois grupos, onde um utilizou a abordagem para verificar o cenário e o outro grupo não a utilizou. Os resultados desse estudo empírico podem ser encontrados em Sousa (2016).

6. Conclusão e Trabalhos Futuros

Neste artigo, foi apresentada uma estratégia para verificar especificações de requisitos através de uma regra de mapeamento que possibilita a transformação equivalente de C&L para statecharts. Com base nesta estratégia, é possível: (1) verificar problemas relacionados a incompletude e inconsistência existentes nos cenários já nas fases iniciais de desenvolvimento do software; (2) possibilitar aos analistas de requisitos uma compreensão mais detalhada do domínio do sistema a ser desenvolvido, corroborando para uma especificação de melhor qualidade; (3) apresentar uma representação mais formal de cenários através de statecharts e (4) mostrar que a nossa abordagem é aplicável a cenários que envolvem paralelismo.

Como trabalhos futuros têm-se: (1) apresentar um processo mais leve de construção de cenários com a técnica C&L; (2) usar os annotations do statechart com o objetivo de ajudar na cobertura de casos de teste para processos desenvolvimento que dependem de LN; (3) realizar uma nova avaliação empírica através de um experimento, com o intuito de confirmar que com o uso da nossa estratégia é possível identificar um maior número de problemas nos cenários do que quem não usou a estratégia.

Agradecimentos. Agradecemos a FACEPE (Fundação de Amparo à Ciência e Tecnologia do Estado de Pernambuco) por financiar esta pesquisa.

References

Santander, A. F. V. (2002) “Integrando Modelagem Organizacional com Modelagem Funcional”. Tese de doutorado (Pós-graduação em ciência da computação) - Universidade Federal de Pernambuco, Recife – PE.

- Kotonya, G. e Sommerville, I. (1998) "Requirements Engineering: Processes and Techniques". New York, NY, USA: John Wiley & Sons, Inc.
- Sarmiento, E; Leite, J. C. S. do P. e Almentero, E. (2014) "C&L: Generating Model Based Test Cases from Natural Language Requirements Descriptions". In: RET, Karlkrona, Sweden.
- Leite, J. C. S. P.; Hadad, G. D. S.; Doorn, J. H. e Kaplan, G. N. (2000) "A Scenario Construction Process". In: RE Conference. v. 05, pp. 38 – 61.
- Leite, J. C. S. do P. e Franco, A. P. M. (1993) "A Strategy for Conceptual Model Acquisition". In.: IEEE Xplore Digital Library.
- Leite, J.C.S.P.; Rossi, G.; Balaguer; Maiorana, V.; Kaplan, G., Hadad, G. e Oliveros, A. (1997) "Enhancing a requirements baseline with scenarios". In: Proceedings of the Third IEEE International Symposium on Requirements Engineering – RE'97, IEEE Computer Society Press, pp. 44-53.
- Ryser, J. e Glinz, M. (1999) "A Scenario-Based Approach to Validating and Testing Software Systems Using Statecharts". In: 12th International Conference on Software and Systems Engineering and their Applications, ICSSEA. Proceedings: CNAM, Paris, France.
- Harel, D. (1987) "Statecharts: A visual formalism for complex systems". In: Science of Computer Programming, 8, (3), 231-274.
- Sarmiento, E.; Leite, J. C. S. P. e Almentero, E. (2015) "Analysis of Scenarios with Petri-Net Models". In.: SBES, Belo Horizonte – MG.
- Pimentel, J.; Castro, J.; Mylopoulos, J.; Angelopoulos, K. e Souza, V. E. S. (2014) "From Requirements to Statecharts via Design Refinement". In: ACM SAC, Gyeongju, Korea.
- Zorman, L. (1995) "Requirements envisaging by utilizing scenarios (Rebus)". PhD dissertation, University of Southern California, 11.
- Leite, J. C. S. P. e Franco, A. P. M. (1990) "O Uso de Hipertexto na Elicitação Linguagens da Aplicação". In: Anais de IV Simpósio Brasileiro de Engenharia de Software. SBC, Brasil, pp 134–149. 13.
- Vilela, J. F. F. (2015) "On the behavior of context-sensitive systems". In: CibSE - WER (Workshop em Engenharia de Requisitos), Lima – Peru.
- Sousa, S. P. de A. (2016) "Regras de Mapeamento e Heurísticas: Uma Estratégia Paracriar Casos de Teste a Partir de C&L Via Statechart". Dissertação de Mestrado. Universidade Federal de Pernambuco, Brasil, 2016.